**Project 2**
**Due: by 11:59pm on February 21, 2022**

For this project, you will mine text from Shakespeare's famous play **Hamlet**. You will only mine from **Act I**. The corresponding text file can be found inside the same zipped folder.

**What to do?**

1. First mine all the sentences in Act I using regular expressions. Please remove the characters' names. Here is an example:

   Before step1:

   ```
   BARNARDO  Who's there?

   FRANCISCO
   Nay, answer me. Stand and unfold yourself.
   ```

   After step1:

   ```
   Who's there?
   Nay, answer me.
   Stand and unfold yourself.
   ```

   Things to consider:

   a. Note that sentences end with a period (.) or a question mark (?) or an exclamation mark (!). However, sometimes period (.) can be found in the middle of a sentence, for example, Ph.D. , Mr. , Mrs. Dr., etc. An interesting thing to note is that period, when appears at the end of a sentence, is usually also followed by an uppercase letter which is a part of the next sentence. This idea will stay true for most cases in Act I.
   b. Note that you will have to be careful to detect the beginning of a sentence. Sentences cannot include a character's name unless it is a part of the dialog. That means, "BARNARDO  Who's there?" is not accepted, because "BARNARDO" is indicating the speaker's name and not part of the dialog. So "BARNARDO" should not be included when mining the sentence. Only "Who's there?" should be mined. However, "Get thee to bed, Francisco." is a valid sentence and should be included, because "Francisco" is a part of the dialog.
   c. Also note that some sentences start with a single quote ('). These sentences should not be ignored.
   d. The above three are just examples, and more patterns may exist. You are encouraged to observe more patterns and use that knowledge to write a

regular expression. You are advised to examine the text carefully before writing your regular expression.

e. You should be able to identify all sentences in Act I using one regular expression.
f. Note that use of built in Python string methods are discouraged and may not be helpful.

2. Store the mined sentences in a Python list. Print the length of the Python list using code. Each item in the list should be one sentence.

3. Once you are done with this step, identify all the bigrams in which a noun is followed by a verb. For this step, feel free to use the NLTK package. NLTK package has a way to identify parts of speech. Do necessary research for the related code. The following Python code will give you a list of possible parts of speech tags.

```
import nltk
nltk.help.upenn_tagset()
```

   You may note that many of the tags for nouns and verbs share the same prefixes. Use this knowledge to write necessary regular expressions to combine the tags for nouns and verbs.

   Also, please make sure you are not double-counting uppercase and lowercase letters for this step. Use regular expressions in your code to ensure that.

   Finally, ignore any bigram with a count lower than 10.

4. Write necessary code to write all the bigrams and their counts to a text file.

5. Write a report (max 3 pages). The report should explain your regular expression breaking it down into smaller parts. You should also include the ideas you used, the key challenges you faced, and any other relevant information. The report should be in PDF format.

**What to know and what to submit?**

1. This is a solo project. You are encouraged to start early, as it might take unexpectedly long time to write regular expressions.
2. You can write your code in a Jupyter Notebook or in a file with .py extension.
3. Put your Notebook file or .py file, the PDF file, and the text files in a folder. Your code should refer to this folder to read from the given text file and write to the file suggested in step 4.
4. Zip the folder and name it as yourLastName_yourFirstName.zip.
5. Upload the zipped file on Canvas.